

1           CLAIMS

2

3       1. A system comprising:

4           a memory to store a plurality of triangle definition structures corresponding  
5           to a plurality of triangles, wherein each of the plurality of triangle definition  
6           structures includes a set of three vertices for the triangle, and wherein each of the  
7           plurality of triangle definition structures further includes a set of three edges,  
8           wherein each of the three edges corresponds to one of the three vertices, and  
9           wherein each of the three edges is an identification of the next edge that is  
10          encountered when performing a traversal in a particular direction about the  
11          corresponding vertex; and

12           a plurality of modules, coupled to be able to operate on the plurality of  
13          triangle definition structures, where each of the plurality of modules includes one  
14          or more instructions, and wherein each of the plurality of modules, when executed,  
15          manipulates various aspects of one or more of the plurality of triangle definition  
16          structures.

17

18       2. A system as recited in claim 1, wherein the particular direction  
19          comprises a counter-clockwise direction.

20

21       3. A system as recited in claim 1, wherein one of the plurality of  
22          modules comprises an add triangle module that adds a new triangle definition  
23          structure to the plurality of triangle definition structures.

1           4. A system as recited in claim 3, wherein the add triangle module  
2 receives, as an input, a set of three vertices from which to create the new triangle  
3 definition structure.

4

5           5. A system as recited in claim 1, wherein one of the plurality of  
6 modules comprises a remove triangle process, wherein the remove triangle process  
7 removes a triangle definition structure from the plurality of triangle definition  
8 structures.

9

10          6. A system as recited in claim 5, wherein the remove triangle process  
11 further modifies one or more other definition structures of the plurality of triangle  
12 definition structures to account for the triangle definition structure being removed.

13

14          7. A system as recited in claim 1, wherein one of the plurality of  
15 modules comprises a rotate operator that receives an edge of a triangle as an input,  
16 and returns a next edge in the triangle in the counterclockwise direction.

17

18          8. A system as recited in claim 1, wherein one of the plurality of  
19 modules comprises an inverse rotate operator that receives an edge of a triangle as  
20 an input, and returns the next edge in the triangle in the clockwise direction.

1           9. A system as recited in claim 1, wherein one of the plurality of  
2 modules comprises a next edge operator that receives an edge of a triangle as an  
3 input, and returns the next edge from the triangle when rotating in a  
4 counterclockwise direction about a vertex of the triangle that is the origin of the  
5 input edge.

6

7           10. A system as recited in claim 1, wherein one of the plurality of  
8 modules comprises a previous edge operator that receives an edge of a triangle as  
9 an input, and returns the next edge from the triangle when rotating in a clockwise  
10 direction about a vertex of the triangle that is the origin of the input edge.

11

12          11. A system as recited in claim 1, wherein one of the plurality of  
13 modules comprises a same operator that receives an edge of a triangle as an input,  
14 and returns an edge of another triangle that has the same vertices as the input edge.

15

16          12. A system as recited in claim 1, wherein one of the plurality of  
17 modules comprises a origin operator that receives an edge of a triangle as an input,  
18 and returns a vertex that is an origin of the edge.

19

20          13. A system as recited in claim 1, wherein one of the plurality of  
21 modules comprises a destination operator that receives an edge of a triangle as an  
22 input, and returns a vertex that is a destination of the edge.

1           **14.** A system as recited in claim 1, wherein one of the plurality of  
2 modules comprises a right operator that receives an edge of a triangle as an input,  
3 wherein the input edge has an origin vertex, and wherein the right operator returns  
4 another vertex of an edge that shares the origin vertex with the input edge.

5  
6           **15.** A system as recited in claim 1, wherein one of the plurality of  
7 modules comprises a left operator that receives an edge of a triangle as an input,  
8 and returns a vertex that is a destination of the next edge from the triangle when  
9 rotating in a counterclockwise direction about a vertex of the triangle that is the  
10 origin of the input edge.

11  
12          **16.** A system as recited in claim 1, wherein one of the plurality of  
13 modules comprises a get representative edge operator that receives a vertex of a  
14 triangle as an input, and returns a representative edge of the vertex.

15  
16          **17.** A system as recited in claim 1, wherein one of the plurality of  
17 modules comprises a set representative edge operator that receives both an edge of  
18 a triangle and a vertex of the triangle as inputs, and sets the input edge to be the  
19 representative edge of the input vertex.

20  
21          **18.** A system as recited in claim 1, wherein one of the plurality of  
22 modules comprises a set origin operator that receives both an edge of a triangle  
23 and a vertex of the triangle as inputs, and sets a vertex of the triangle  
24 corresponding to an origin of the edge to be the input vertex.

1           **19.** A system as recited in claim 1, wherein one of the plurality of  
2 modules comprises a set next edge operator that receives both a first edge and a  
3 second edge as inputs, and sets one of the edges in the triangle definition structure  
4 for the triangle that the first edge is part of to be the second edge.

5

6           **20.** A system as recited in claim 1, wherein one of the plurality of  
7 modules comprises a make edge operator that receives both a first vertex and a  
8 second vertex as inputs, and generates two triangle definition structures, wherein  
9 the two triangle definition structures have adjacent edges between the first vertex  
10 and the second vertex.

11

12          **21.** A system as recited in claim 1, wherein one of the plurality of  
13 modules comprises a make edge operator that receives both a first vertex and a  
14 second vertex as inputs, and generates two triangle definition structures to be  
15 added to the plurality of triangle definition structures, wherein each of the two  
16 triangle definition structures identifies, as the set of three vertices of the  
17 corresponding triangles, the input first vertex, the input second vertex, and a third  
18 vertex that is a boundary vertex.

19

20          **22.** A system as recited in claim 21, wherein each of the two triangle  
21 definition structures further identifies, as the set of three edges for the  
22 corresponding triangles, edges of the other triangle.

1        23. A system as recited in claim 1, wherein one of the plurality of  
2 modules comprises a splice operator that receives both a first edge and a second  
3 edge as inputs, alters the set of edges of the triangle definition structure of the  
4 triangle including the first edge, and alters the set of edges of the triangle  
5 definition structure of the triangle including the second edge.

6  
7        24. A system as recited in claim 1, wherein one of the plurality of  
8 modules comprises a splice operator that receives both a first edge and a second  
9 edge as inputs, and wherein the splice operator:

10              identifies a third edge as a next edge, from a first triangle including the first  
11 edge, when rotating in a counterclockwise direction about a vertex of the first  
12 triangle that is the origin of the first edge;

13              identifies a fourth edge as a next edge, from a second triangle including the  
14 second edge, when rotating in a counterclockwise direction about a vertex of the  
15 second triangle that is the origin of the second edge;

16              identifies a fifth edge as a next edge in the first triangle in the  
17 counterclockwise direction;

18              identifies a sixth edge as a next edge in the second triangle in the  
19 counterclockwise direction;

20              identifies a seventh edge as a next edge in the first triangle in the clockwise  
21 direction;

22              identifies an eighth edge as a next edge in the second triangle in the  
23 clockwise direction;

sets, as a next edge that would be encountered when rotating in a  
counterclockwise direction from the first edge about the origin of the first edge,  
the fourth edge;

sets, as a next edge that would be encountered when rotating in a  
counterclockwise direction from the second edge about the origin of the second  
edge, the third edge;

sets, as a next edge that would be encountered when rotating in a  
counterclockwise direction from the fifth edge about the origin of the fifth edge,  
the eighth edge; and

sets, as a next edge that would be encountered when rotating in a  
counterclockwise direction from the sixth edge about the origin of the sixth edge,  
the seventh edge.

**25.** A system as recited in claim 24, wherein one of the plurality of  
modules comprises a swap operator that receives a particular edge of a triangle as  
an input, and wherein the swap operator:

identifies a ninth edge as a next edge, from the input particular edge, when  
rotating in a counterclockwise direction about the origin of the input particular  
edge;

identifies a tenth edge as a next edge in the input triangle in the  
counterclockwise direction from the input particular edge;

identifies a destination vertex of the ninth edge;

identifies a destination vertex of the tenth edge;

sets an origin of the ninth edge to be the destination vertex of the tenth  
edge;

1 sets an origin of the tenth edge to be the destination vertex of the ninth  
2 edge;

3 invokes the splice operator and uses, as inputs to the splice operator, the  
4 ninth edge and the input particular edge;

5 identifies an eleventh edge having the same vertices as the tenth edge; and

6 again invokes the splice operator and uses, as inputs to the splice operator,  
7 the ninth edge and the eleventh edge.

8

9       **26.** A system as recited in claim 1, wherein one of the plurality of  
10 modules comprises a swap operator that receives an edge of a triangle as an input,  
11 and returns an opposite diagonal of a quadrilateral corresponding to the input  
12 edge.

13

14       **27.** A computer readable medium having stored thereon a data structure  
15 for defining a triangle, the data structure comprising:

16           a first data field identifying a set of three vertices for the triangle; and

17           a second data field identifying a set of three edges, wherein each of the  
18 three edges corresponds to one of the three vertices, and wherein each of the three  
19 edges is derived from the first data field by determining the next edge that is  
20 encountered when performing a traversal about the corresponding vertex identified  
21 in the first data field.

1           **28.** A data structure as recited in claim 27, wherein the traversal about  
2 the corresponding vertex is a counter-clockwise traversal about the corresponding  
3 vertex.

4

5           **29.** A data structure as recited in claim 27, wherein each vertex of the  
6 set of three vertices includes a set of values representing the location of the vertex  
7 and an identification of a representative triangle edge corresponding to the vertex.

8

9           **30.** A data structure as recited in claim 27, wherein each edge of the set  
10 of three edges includes:

11                 an identifier of another triangle, the other triangle being the triangle that the  
12 next edge is part of; and

13                 a position index indicating a position of the edge in the other triangle.

14

15           **31.** A data structure as recited in claim 30, wherein each edge of the set  
16 of three edges further includes a flip indicator identifying which direction the  
17 edges are to be viewed in.

18

19           **32.** A method comprising:

20                 creating a triangle based on a pair of triples;

21                 wherein the first triple is a set of three vertices for the triangle; and

22                 wherein the second triple is a set of three edges, wherein each of the three  
23 edges corresponds to one of the three vertices, and wherein each of the three edges  
24 is an identification of the next edge that is encountered when performing a  
25 traversal about the corresponding vertex.

1  
2       **33.** A method as recited in claim 32, wherein the traversal about the  
3 corresponding vertex is a counter-clockwise traversal about the corresponding  
4 vertex.  
5

6       **34.** A method as recited in claim 32, wherein each vertex of the set of  
7 three vertices includes a set of values representing the location of the vertex and an  
8 identification of a representative triangle edge corresponding to the vertex.  
9

10      **35.** A method as recited in claim 32, wherein each edge of the set of  
11 three edges includes:  
12           an identifier of another triangle, the other triangle being the triangle that the  
13 next edge is part of; and  
14           a position index indicating a position of the edge in the other triangle.  
15

16      **36.** A method as recited in claim 35, wherein each edge of the set of  
17 three edges further includes a flip indicator identifying which direction the edges  
18 are to be viewed in.  
19

20      **37.** One or more computer-readable memories containing a computer  
21 program that is executable by a processor to perform the method recited in claim  
22 32.  
23  
24  
25

1           **38.** One or more computer-readable media having stored thereon a  
2 plurality of instructions that, when executed by one or more processors of a  
3 computer, causes the one or more processors to perform the following acts:

4                 accessing a structure defining a triangle; and  
5                 identifying, by accessing a particular portion of the structure, a next edge  
6 that is encountered when performing a traversal in a particular direction about one  
7 vertex of the triangle.

8  
9           **39.** One or more computer-readable media as recited in claim 38,  
10 wherein the particular direction comprises a counter-clockwise direction.

11  
12           **40.** One or more computer-readable media as recited in claim 38,  
13 wherein the structure defining the triangle includes a set of three vertices, and  
14 wherein each vertex of the set of three vertices includes a set of values  
15 representing the location of the vertex and an identification of a representative  
16 triangle edge corresponding to the vertex.

17  
18           **41.** One or more computer-readable media as recited in claim 38,  
19 wherein the plurality of instructions further cause the one or more processors to  
20 perform the following acts:  
21

22                 identifying, by accessing another portion of the structure, a next edge that is  
23 encountered when performing a traversal in the particular direction about a second  
24 vertex of the triangle; and  
25

1 identifying, by accessing another portion of the structure, a next edge that is  
2 encountered when performing a traversal in the particular direction about a third  
3 vertex of the triangle.

4

5       **42.** A method for adding a triangle to a triangular mesh, the triangle  
6 having three vertices and three edges, the method comprising:

7             receiving an indication of the three vertices of the triangle;  
8             checking whether any of the three edges of the triangle already exist in the  
9 triangular mesh;

10            if any of the three edges of the triangle already exist in the triangular mesh,  
11 then checking connectivity of the mesh, determining if any connectivity of the  
12 triangular mesh needs to be changed, and changing connectivity of the triangular  
13 mesh as necessary to accommodate the triangle;

14            creating additional edges for the triangle as necessary until two edges of the  
15 triangle exist; and

16            connecting free ends of the two edges with an additional edge to create the  
17 triangle.

18

19       **43.** A method as recited in claim 42, further comprising setting a  
20 representative edge for each vertex of the triangle as necessary.

21

22       **44.** A method as recited in claim 42, further comprising setting a  
23 representative edge for each vertex of the triangle.

1           **45.** A method as recited in claim 42, wherein changing connectivity of  
2 the triangular mesh as necessary to accommodate the triangle comprises altering  
3 an edge identifier corresponding to another triangle of the triangular mesh to  
4 identify an edge of the triangle as a next edge that is encountered when performing  
5 a traversal in a particular direction about a vertex of the other triangle.

6

7           **46.** One or more computer-readable memories containing a computer  
8 program that is executable by a processor to perform the method recited in claim  
9 42.

10

11           **47.** A method for removing a triangle from a triangular mesh, wherein  
12 each triangle in the triangular mesh includes an identification of three edges, and  
13 wherein each of the three edges is an identification of the next edge that is  
14 encountered when performing a traversal about a vertex corresponding to the edge,  
15 the method comprising:

16           receiving an identifier of the triangle to be removed;

17           updating a representative edge of each vertex of the triangle so that the  
18 representative edge is not an edge of the triangle; and

19           for each edge of the triangle, removing the edge and changing the  
20 connectivity of other triangles in the triangular mesh so that any other triangle  
21 having an identification of a next edge that is an edge of the triangle being  
22 removed has the identification changed to another edge of the triangular mesh.

1           **48.** A method as recited in claim 47, wherein the updating comprises,  
2 for each vertex:

3                 identifying an edge of the triangle, wherein the edge has an origin at the  
4 vertex;

5                 identifying, based on the edge, a previous edge that is the next edge from  
6 the edge when rotating in a clockwise direction about the vertex;

7                 identifying a right vertex based on the previous edge, wherein the right  
8 vertex is another vertex of another edge that also has the origin as one of its  
9 vertices;

10                 checking whether the right vertex is a boundary vertex; and

11                 if the right vertex is not a boundary vertex, then setting the representative  
12 edge for the vertex to be the previous edge.

13  
14           **49.** A method as recited in claim 48, wherein for each vertex if the right  
15 vertex is a boundary vertex, then the updating further comprises:

16                 identifying, based on the previous edge, another previous edge that is the  
17 next edge from the previous edge when rotating in a clockwise direction about the  
18 vertex;

19                 identifying, based on the edge, a subsequent edge that is the next edge from  
20 the edge when rotating in a counter-clockwise direction about the vertex;

21                 checking whether the other previous edge and the subsequent edge are the  
22 same edges;

23                 if the other previous edge and the subsequent edge are the same edges, then  
24 setting the representative edge for the vertex to be empty; and

1 if the other previous edge and the subsequent edge are not the same edges,  
2 then identifying, based on the other previous edge, a third previous edge that is the  
3 next edge from the other previous edge when rotating in a clockwise direction  
4 about the vertex, and setting the representative edge for the vertex to be the third  
5 previous edge.

6

7       **50.** A method as recited in claim 47, further comprising selectively  
8 invoking a make edge operator, wherein the make edge operator receives both a  
9 first vertex and a second vertex as inputs, and generates two triangles, wherein  
10 each of the two triangles identifies, as a set of three vertices of the corresponding  
11 triangles, the input first vertex, the input second vertex, and a third vertex that is a  
12 boundary vertex, and wherein each of the two triangles further identifies, as a set  
13 of three edges for the corresponding triangles, edges of the other triangle.

14

15       **51.** One or more computer-readable media having stored thereon a  
16 plurality of instructions to manage a triangular mesh, wherein the plurality of  
17 instructions, when executed by one or more processors of a computer, causes the  
18 one or more processors to perform the following acts:

19              implementing a make edge operator that receives both a first vertex and a  
20 second vertex as inputs, and generates two triangles, wherein the two triangles  
21 have adjacent edges between the first vertex and the second vertex;

22              implementing a splice operator that receives both a first edge and a second  
23 edge as inputs, alters connectivity of a triangle including the first edge, and alters  
24 connectivity of a triangle including the second edge; and

1       implementing a swap operator that receives a particular edge of a triangle  
2       as an input, and returns an opposite diagonal of a quadrilateral corresponding to  
3       the input edge.

4

5       **52.** One or more computer-readable media as recited in claim 51,  
6       wherein the instructions that cause the one or more processors to implement the  
7       make edge operator include instructions that cause the one or more processors to  
8       perform the following acts:

9             receiving both the first vertex and the second vertex as inputs; and  
10            generating the two triangles, wherein each of the two triangles identifies, as  
11            a set of three vertices of the corresponding triangle, the input first vertex, the input  
12            second vertex, and a third vertex that is a boundary vertex, and wherein each of  
13            the two triangles further identifies, as a set of three edges for the corresponding  
14            triangle, edges of the other triangle.

15

16       **53.** One or more computer-readable media as recited in claim 51,  
17       wherein the instructions that cause the one or more processors to implement the  
18       splice operator include instructions that cause the one or more processors to  
19       perform the following acts:

20             receiving both the first edge and the second edge as inputs;  
21            identifying a third edge as a next edge encountered, from the first edge,  
22       when rotating in a counterclockwise direction about a vertex that is the origin of  
23       the first edge;

1 identifying a fourth edge as a next edge encountered, from the second edge,  
2 when rotating in a counterclockwise direction about a vertex that is the origin of  
3 the second edge;

4 identifying a fifth edge as a next edge, in the triangle including the first  
5 edge, in the counterclockwise direction;

6 identifying a sixth edge as a next edge, in the triangle including the second  
7 edge, in the counterclockwise direction;

8 identifying a seventh edge as a next edge, in the triangle including the first  
9 edge, in the clockwise direction;

10 identifying an eighth edge as a next edge, in the triangle including the  
11 second edge, in the clockwise direction;

12 setting, as a next edge that would be encountered when rotating in a  
13 counterclockwise direction from the first edge about the origin of the first edge,  
14 the fourth edge;

15 setting, as a next edge that would be encountered when rotating in a  
16 counterclockwise direction from the second edge about the origin of the second  
17 edge, the third edge;

18 setting, as a next edge that would be encountered when rotating in a  
19 counterclockwise direction from the fifth edge about the origin of the fifth edge,  
20 the eighth edge; and

21 setting, as a next edge that would be encountered when rotating in a  
22 counterclockwise direction from the sixth edge about the origin of the sixth edge,  
23 the seventh edge.

1       **54.** One or more computer-readable media as recited in claim 53,  
2 wherein the instructions that cause the one or more processors to implement the  
3 swap operator include instructions that cause the one or more processors to  
4 perform the following acts:

5             receiving the particular edge of the triangle as an input;

6             identifying a ninth edge as a next edge, from the input particular edge,  
7 when rotating in a counterclockwise direction about the origin of the input  
8 particular edge;

9             identifying a tenth edge as a next edge in the input triangle in the  
10 counterclockwise direction from the input particular edge;

11            identifying a destination vertex of the ninth edge;

12            identifying a destination vertex of the tenth edge;

13            setting an origin of the ninth edge to be the destination vertex of the tenth  
14 edge;

15            setting an origin of the tenth edge to be the destination vertex of the ninth  
16 edge;

17            invoking the splice operator and using, as inputs to the splice operator, the  
18 ninth edge and the input particular edge;

19            identifying an eleventh edge having the same vertices as the tenth edge; and

20            again invoking the splice operator and using, as inputs to the splice  
21 operator, the ninth edge and the eleventh edge.

1       **55.** One or more computer-readable media as recited in claim 51,  
2 wherein the plurality of instructions further cause the one or more processors to  
3 perform the following acts:

4             selectively invoking the make edge operator, the splice operator, and the  
5 swap operator to add triangles to the triangular mesh; and

6             selectively invoking the make edge operator, the splice operator, and the  
7 swap operator to remove triangles from the triangular mesh.

8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25